

## Contents

<b>1 Routine/Function Prologues</b>	<b>2</b>
1.0.1 retgdas.F90 (Source File: retgdas.F90) . . . . .	2
1.0.2 interp_gdas (Source File: retgdas.F90) . . . . .	4

## 1 Routine/Function Prologues

### 1.0.1 retgdas.F90 (Source File: retgdas.F90)

Defines forcing parameters, retrieves the fields using calls to getgb, and interpolates the fields to LDAS specifications

#### REVISION HISTORY:

```

14 Dec 2000: Urszula Jambor; Rewrote geteta.f to use GDAS forcing in GLDAS
15 Mar 2001: Jon Gottschalck; Added additional parameters and octets in
              which to search in GRIB files
01 Jun 2001: Urszula Jambor; Added option to get forcing from different
              files (F00 instantaneous and F06 six hour means)
29 Jan 2003: Urszula Jambor; Rewrote code, uses GETGB call to replace
              ungribgdas. Interpolation now occurs in interp_gdas.
              Using GETGB avoids problems with the Oct2002 GDAS
              grid update.
12 Nov 2003: Matt Rodell; Check to make sure input file exists before
              opening and thereby creating a new, empty file.
14 Nov 2003: Matt Rodell; Ensure lugb varies in call to baopen

```

#### INTERFACE:

```
subroutine retgdas( order, ld, gindex, name, nameF06, F00flag,ferror,try )
```

#### USES:

```

use lis_module      ! LDAS non-model-specific 1-D variables
use time_manager
use baseforcing_module, only: glbdata1, glbdata2
use gdasdomain_module, only : gdasdrv
implicit none

```

#### ARGUMENTS:

```

type (lisdec) ld
integer :: gindex(ld%d%lnc, ld%d%lnr)
integer :: order    ! 1 indicates lesser interp. bdry, 2 indicates greater
character(len=80) :: name, nameF06
integer :: F00flag  ! if 1, need for data from 2 files (name, nameF06)
integer :: ferror   ! set to zero if there's an error
integer :: try

```

#### CONTENTS:

```

varfield = 0.0
ngdas = (gdasdrv%ncold*gdasdrv%nrold)
!-----
! Set the GRIB parameter specifiers
!-----
if (get_nstep() .eq. 0) then

```

```

pds5 = (/011,051,204,205,033,034,001,059,214,084,144,144, 011,011, 065/) !parameter
pds7 = (/002,002,000,000,010,010,000,000,000,000,010,2760,010,2760,000/) !htlev2
nforce = gdasdrv%nmif
else
  pds5 = (/ 011,051,204,205,033,034,001,059,214,084 /) !parameter
  pds7 = (/ 002,002,000,000,010,010,000,000,000,000 /) !htlev2
  nforce = 10 ! for now
endif

ferror = 1
!-----
! if there's a problem then ferror is set to zero
!-----
iv = 0
errorflag = 0
endloop = 0

do
  if ( endloop == 1 ) exit
  iv = iv+1

  if ( (F00flag > 0) .and. &
      (iv==3.or.iv==4.or.iv==8.or.iv==9.or.iv==10) ) then
    fname = nameF06
  else
    fname = name
  end if
  inquire (file=fname, exist=file_exists)
  if (file_exists) then
!-----
! Set up to open file and retrieve specified field
!-----
    lugb = iv +try
    j = 0
    jpds = -1
    jpds(5) = pds5(iv)
    jpds(7) = pds7(iv)
    call baopen(lugb,fname,iret)
    if(iret==0) then
      call getgb(lugb,lubi,ngdas,j,jpds,jgds,kf,k,kpds,kgds,lb,f,gbret)
    else
      gbret = 99
    endif
    call baclose(lugb,jret)
  else
    ferror = 0
    return
  endif

```

```

!-----
! If field successfully retrieved, interpolate to LIS domain
!-----

      if (gbret==0) then
          call interp_gdas(kpds,kgds,ngdas,f,lb,ld%d%kgds, &
                           ld%d%lnc,ld%d%lnr,varfield)
      else
          errorflag = 1
      endif

      if ( errorflag == 1 ) then
          endloop = 1
          ferror = 0
      else
          do c =1, ld%d%lnc
              do r = 1, ld%d%lnr
                  if (gindex(c,r).ne. -1) then
                      if ( order == 1 ) then
                          glbdata1(iv,gindex(c,r)) = varfield(c,r)
                      else
                          glbdata2(iv,gindex(c,r)) = varfield(c,r)
                      end if
                  endif
              end do
          end do
      end if

      if ( errorflag == 1 ) then
          print *, 'Could not find correct forcing parameter in file',name
      end if
      if ( iv == nforce ) endloop = 1
    end do
    return

```

### 1.0.2 interp\_gdas (Source File: retgdas.F90)

This subroutine interpolates a given GDAS field to the LIS domain. Special treatment for some initialization fields. Code based on old ungribgdas.f

**INTERFACE:**

```
subroutine interp_gdas(kpds,kgds,ngdas,f,lb,lis_gds,nc,nr, &
                       varfield)
```

**USES:**

```
use def_ipMod, only : w110,w120,w210,w220,n110,n120,n210,n220,&
```

```
rlat0,rlon0
use gdasdomain_module, only : mi
```

## CONTENTS:

```
!-----
! Setting interpolation options (ip=0,bilinear)
! (km=1, one parameter, ibi=1,use undefined bitmap
! (needed for soil moisture and temperature only)
! Use budget bilinear (ip=3) for precip forcing fields
!-----

nglis = nc*nr
if (kpds(5)==59 .or. kpds(5)==214) then
    ip=3
    ipopt(1)=-1
    ipopt(2)=-1
    km=1
    ibi=1
else
    ip=0
    do i=1,20
        ipopt(i)=0
    enddo
    km=1
    ibi=1
endif
!-----
! Initialize output bitmap. Important for soil moisture and temp.
!-----
lo = .true.
!-----
! Interpolate to LIS grid
!-----
call polates0 (lis_gds,ibi,lb,f,ibo,lo,lis1d,mi,&
    rlat0, rlon0,w110,w120,w210,w220,n110,n120,n210,n220,iret)
!-----
! Create 2D array for main program. Also define a "soil" mask
! due to different geography between GDAS & LDAS. For LDAS land
! points not included in GDAS geography dataset only.
!-----
count = 0
do j = 1, nr
    do i = 1, nc
        varfield(i,j) = lis1d(i+count)
        geogmask(i,j) = lo(i+count)
    enddo
    count = count + nc
enddo
!-----
```

```
! Save air tempertaure interpolated field for later use in
! initialization of soil temp where geography differs
! between GDAS and LDAS
!-----
if (kpds(5) .eq. 11 .and. kpds(6) .eq. 105) then
  do i = 1, nc
    do j = 1, nr
      geogtemp(i,j) = varfield(i,j)
    enddo
  enddo
endif
```